regulators: Learning to play with others

Mark Brown, ELC 2015, San Jose

Introduction

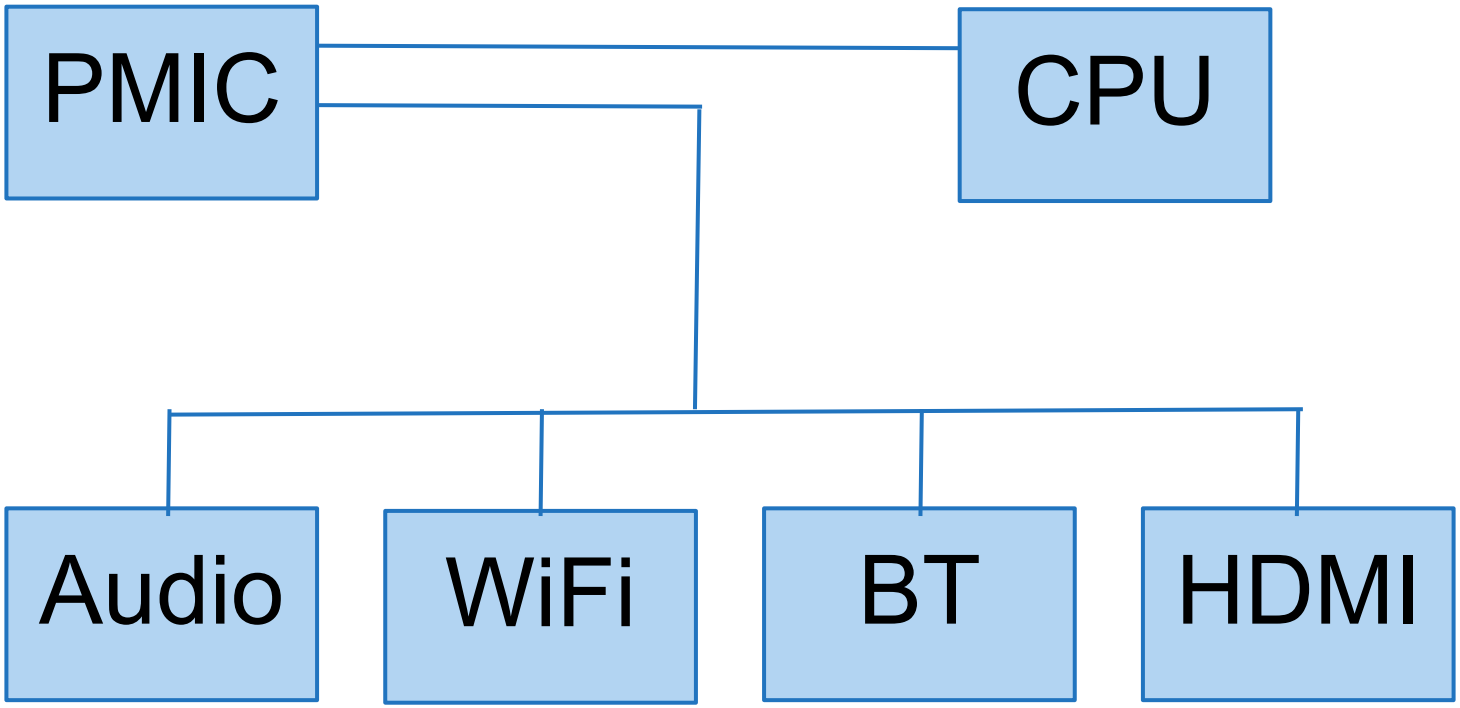
- Regulator API overview
- Modern systems
- Non-regulator solutions
- Microcontroller interfacing
- Suspend/idle integration
- Future work

What is a regulator?

- For Linux we mean voltage regulators
- Takes an input supply, produces a target voltage
- Many different kinds
  - LDO
  - DCDC
  - Boost
- All very similar to software
  - Enable/disable
  - Set voltage
  - Set performance requirements
  - Typically I2C or SPI devices grouping several regulators

- Current regulators do exist, not relevant here

How does it fit into a system?



Why do regulators need drivers?

- Power saving
- Hardware interfacing
  - MMC
- Fix hardware defaults

regulator API - regulator devices

- Drivers register as devices as normal
- Provide set/get operations
  - Enable
  - Voltage
  - Set performance characteristics
- Provide parameters
  - Voltage ranges
  - Time to implement changes
- Standard regmap operations provided
  - Many regulators need only data

regulator API - consumer devices

- Request supply using device side supply name
  - Special interface if supply might be missing
- Read status
- Request configuration
  - Range based interface for voltages
- Notifications provided when configuration changes
- Details of regulator hidden

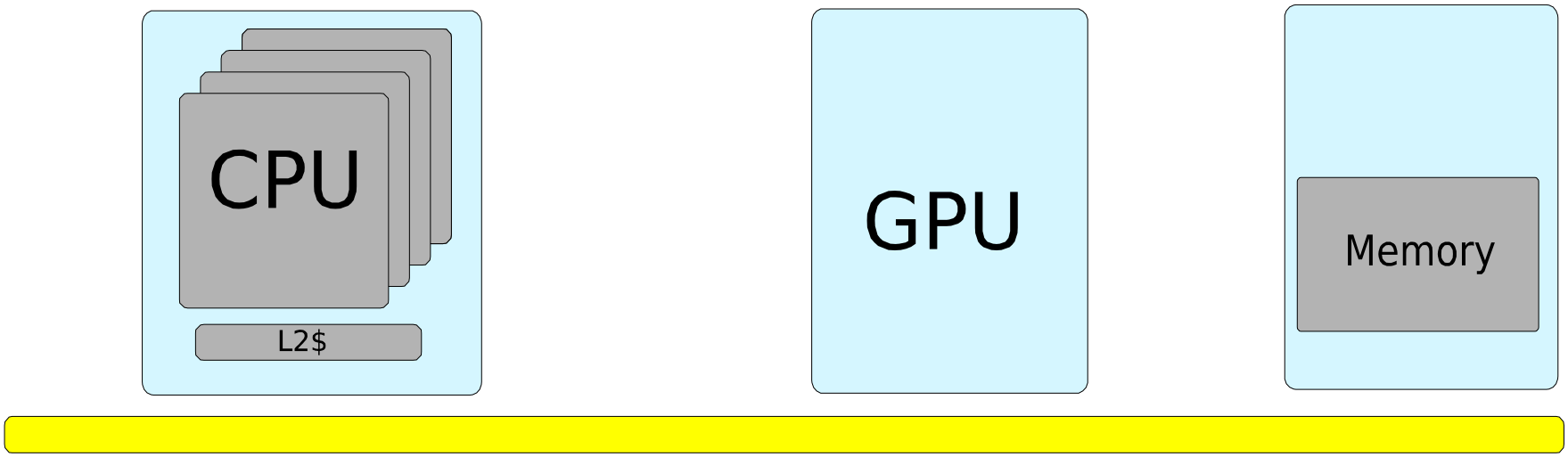
regulator API - system integration

- Firmware or board file maps regulators to devices
- Explicitly says what operations are allowed
  - Range of voltages to set
  - Supported operating modes
  - If supply can be turned off
- Default behaviour is read only
  - Any problems due to system integration!
- Core applies settings from consumers within constraints
  - Combines requests from consumers
  - Settings may not take effect due to other devices or constraints
- Kernel knows exact hardware state at all times

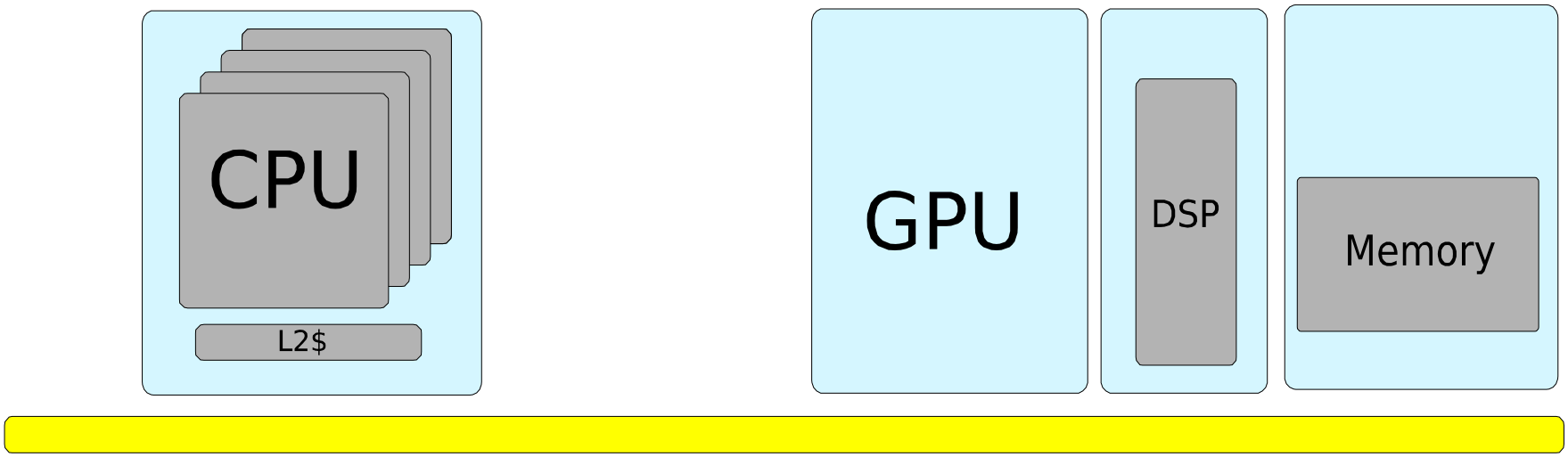
Suspend mode configuration

- Typically handled with hard coded configuration
- Sometimes Linux needs to tweak setup for suspend
  - DT bindings

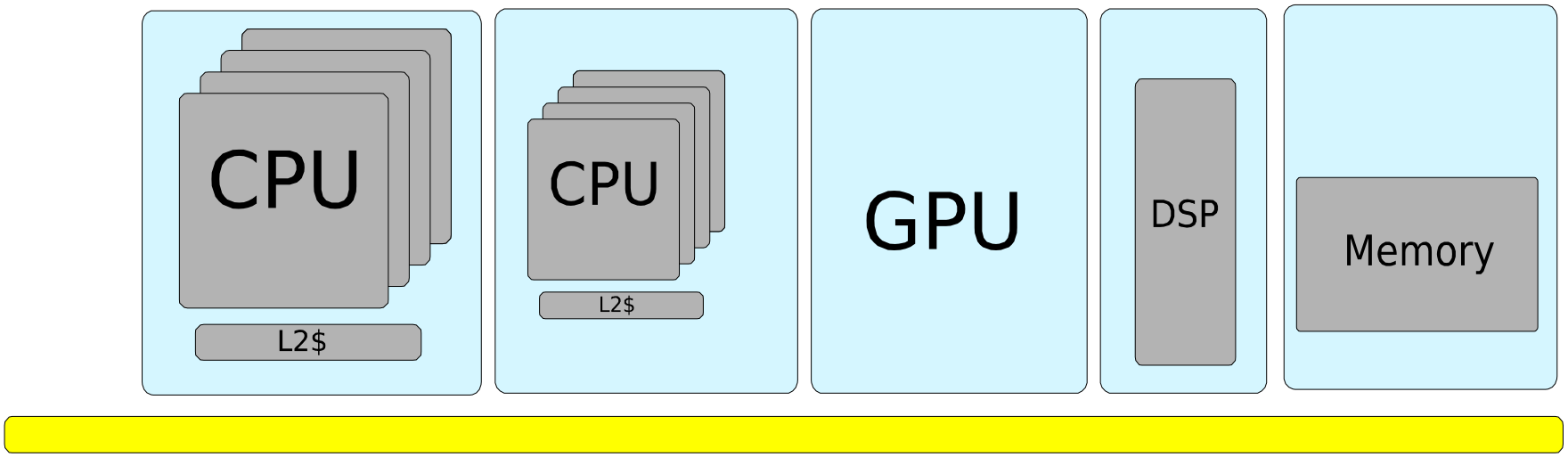
Modern system design



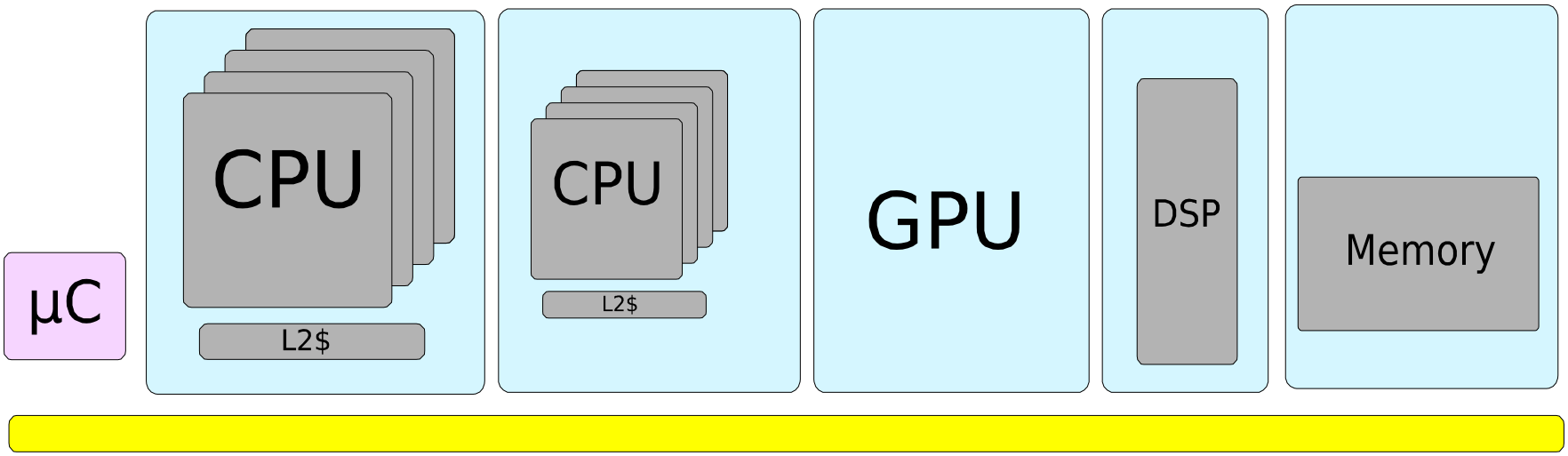
Modern system design



Modern system design



Modern system design



Motivation

- Suspending main AP no longer fixed process
  - Multi-cluster systems
  - Runtime configurable functionality
  - Blurring of distinction between suspend and idle
- Other processors running while main AP is suspended
  - Baseband
  - WiFi
  - Always on sensor monitoring
- Fine grained power optimization
  - Tuning for individual chip characteristics
  - Often need real time response
- Security considerations

Ultra simplified model - ACPI

- Completely hide power control details
  - OS provides device on/off information
  - System integration done in firmware
- Great for servers/laptops
- Limiting for mobile
  - Fine grained power control
  - Too much hardware variation
  - Schedules too tight

Mixed model - hidden subsystem

- Key subsystems hidden from OS
  - Normally CPUs
  - Any OS control via higher level interfaces

- May use some regulators on a shared chip
  - regulator_get_hardware_vsel_register()
  - regulator_list_hardware_vsel()

Mixed model - visible subsystem

- Microcontroller arbitrates between users
  - Core SoC supplies
  - Supplies shared with external components
- Ideally microcontroller offers regulator API interface
  - Zero effort?

System mode mapping

- Linux idea of system state can diverge from hardware
- Qualcomm RPM - two modes for Linux system active
  - Active - Linux running
    - CPU supplies
  - Idle - Linux in idle
    - WiFi

Mode specific configuration

- Generic devices get mapped in DT
  - Default is all modes that apply outside of suspend

- Provide interface for drivers that know about modes
  - regulator_set_voltage(regulator, mode, min, max);
  - regulator_set_voltage_mode(regulator, mode, min, max);

Abstracted settings

- Some microcontrollers provide abstract modes
  - Essentially OPPs
- Can be hidden subsystem model
- Simple to handle for platform specific devices
  - regulator_set_mode() or equivalent
- Need mapping mechanism for more generic devices
  - Yes, people do this

Next steps

- Confirm designs with real systems
- Upstream it

- Improve support for specifying voltages by tolerance
- Support for resolving dependency loops between PMICs

Thanks

- Qualcomm
- Bjorn Andersson
- Stephen Boyd
- Doug Anderson
- Javier Martinez Canillas
- Kevin Hilman

More about Linaro Connect: http://connect.linaro.org
More about Linaro: http://www.linaro.org/about/
More about Linaro engineering: http://www.linaro.org/engineering/
Linaro members: www.linaro.org/members